Characterizing, Predicting, and Handling Web Search Queries That Match Very Few or No Results

Erdem Sarigil
Computer Engineering Department, Bilkent University, Ankara, Turkey. E-mail: esarigil@cs.bilkent.edu.tr

Ismail Sengor Altingovde
Computer Engineering Department, Middle East Technical University, Ankara, Turkey. E-mail: altingovde@ceng.metu.edu.tr

Roi Blanco
University of A Coruña, A Coruña, Spain. E-mail: rblanco@udc.es

B. Barla Cambazoglu
Yahoo! Labs, Barcelona, Spain. E-mail: barla@yahoo-inc.com

Rifat Ozcan
Ntent, Barcelona, Spain. E-mail: rozcan@ntent.com

Özgür Ulusoy
Computer Engineering Department, Bilkent University, Ankara, Turkey. E-mail: oulusoy@cs.bilkent.edu.tr

A non-negligible fraction of user queries end up with very few or even no matching results in leading commercial web search engines. In this work, we provide a detailed characterization of such queries and show that search engines try to improve such queries by showing the results of related queries. Through a user study, we show that these query suggestions are usually perceived as relevant. Also, through a query log analysis, we show that the users are dissatisfied after submitting a query that match no results at least 88.5% of the time. As a first step towards solving these no-answer queries, we devised a large number of features that can be used to identify such queries and built machine-learning models. These models can be useful for scenarios such as the mobile- or meta-search, where identifying a query that will retrieve no results at the client device (i.e., even before submitting it to the search engine) may yield gains in terms of the bandwidth usage, power consumption, and/or monetary costs. Experiments over query logs indicate that, despite the heavy skew in class sizes, our models achieve good prediction quality, with accuracy (in terms of area under the curve) up to 0.95.

Introduction

In today's highly competitive search market, users who are frustrated with not finding the information they seek may abandon their search sessions and switch to another search engine, resulting in losses in the revenue and brand loyalty of a search engine. Indeed, some studies report that almost half of the users switch between search engines at least once per month (White, Richardson, Bilenko, & Heath, 2008; White & Dumais, 2009). According to these studies, more than half of the users state dissatisfaction with search results as the main reason for switching to another search engine.

Dissatisfaction with search results is primarily a consequence of the search engine's inability to surface relevant or useful information. Despite the advances in effectiveness of web search engines, a tangible portion of web queries remain unsolved even by the major web search engines

Additional Supporting Information may be found in the online version of this article.

B. Barla Cambazoglu is currently affiliated with Ntent, Barcelona, Spain.

Received August 27, 2014; revised July 16, 2017; accepted August 7, 2017

 DOI: 10.1002/asi.23955

(Zaragoza, Cambazoglu, & Baeza-Yates, 2010). What makes a search query difficult for a search engine is still open to debate. However, in general, it is agreed that non-navigational, infrequent, or long queries tend to be relatively more difficult (Balasubramanian, Kumaran, & Carvalho, 2010; Downey, Dumais, & Horvitz, 2007; Zaragoza et al., 2010).

In this paper, we focus on a specific subset of difficult queries: those that match very few or no answers when they are issued to a web search engine. Handling this particular subset of queries is important for web search engines because, in general, small answer sets are likely to fail to satisfy users' information needs entirely. In practice, there are many reasons that may lead to a few- or no-answer query (FNAQ). These include the presence of typos that make the interpretation of the query difficult, the inability to match the terms in the query with the text content (e.g., due to a very uncommon query term), or simply the lack of useful content in the search index (e.g., when the query seeks an unpopular webpage that is not yet crawled and indexed by the search engine).

We believe that having a thorough investigation of FNAQs is important, as this may fuel the research on solving such queries, eventually leading to improvements in search result quality as well as user satisfaction and providing significant benefits to commercial web search engines. To the best of our knowledge, so far no previous work in the literature investigated FNAQs in a real web search setting. To fill this gap, our work makes the following three contributions. First, we investigate the result retrieval and query suggestion mechanisms employed by the current web search engines to solve FNAQs. We show that, although these mechanisms improve the result quality of FNAQs, the problem is far from being completely solved. Second, we investigate an extreme case of FNAQs: queries for which no result can be retrieved by the search engine. We refer to such queries as no-answer queries (NAQs) and make the first attempt to characterize such queries through a user study and a quantitative analysis. Our analysis reveals that around 0.1% of the unique queries (from a large query log excerpt) are NAQs, which would constitute a small yet probably non-negligible number of queries for a commercial search engine that may be receiving billions of unique queries per day. Third, we build a machine-learning model to predict the NAQs observed in a real-life query log obtained from Yahoo! Web Search. To motivate our model, we present several use case scenarios where early prediction of NAQs may be useful, such as mobile web search and meta-/federated search. We evaluate our prediction model under different assumptions and demonstrate the feasibility of predicting NAQs.

The rest of the paper is organized as follows. We begin with a literature review in the Related Work section, and then provide an overview of our research in the Research Objectives and Findings section. In the Sampling FNAQs section, we describe our query set used throughout this work and in Handling FNAQs section, we analyze the mechanisms employed by search engines to handle FNAQs. The following two sections, Characterizing NAQs and Predicting NAQs, are devoted to the characterization and prediction of NAQs. Finally, we discuss a number of techniques that may be potentially useful in solving NAQs in the Discussion on Solving NAQs section, and then summarize our findings and point out some future work in the Conclusion section.

Related Work

To the best of our knowledge, no prior work in the literature has focused on characterizing FNAQs or the mechanisms employed by general-purpose web search engines to handle them (while search failures have recently been a topic of interest for more restricted scenarios, such as academic search [Li, Schijvenaars, & de Rijke, 2017]). However, there are earlier studies in closely related topics, such as query classification, query reformulation, and difficult query analysis. Herein, we provide a brief survey of these studies.

Query Classification

In earlier studies, web search queries were categorized and/or classified based on a list of topics, users' search goals, or user interaction. For instance, Beitzel, Jensen, Chowdhury, Grossman, and Frieder, (2004) analyzed the temporal trends for queries that are topically categorized by human editors. Two particular studies attempted to classify long and rare web queries: Broder et al. (2007) exploited retrieved query results for classification and aimed to improve the selection of advertisements for rare queries, while Bailey, White, Liu, and Kumaran, (2010) classified rare queries by matching them against previously seen classified queries.

Query Reformulation

Users are known to refine or reformulate their queries when they are not satisfied with the query results. In Jansen, Spink, and Koshman (2007), analysis of a large query log revealed that almost half of the users (46%) reformulate their queries. Query reformulation can be performed in different ways: the user might replace one or more terms in the query with others, generalize the query by removing a term, or specialize by adding more terms. Spelling correction may be considered as a form of query reformulation. It was shown that around 10–15% of search queries contain spelling errors (Cucerzan & Brill, 2004). Spelling correction was found to be more difficult in the web environment due to the diversity of terms that require special solutions (Cucerzan & Brill, 2004).

Difficult Query Analysis

Carmel, Yom-Tov, Darlow, and Pelleg (2006) analyzed the reasons that make a query difficult. They found that, when the distance of queries and relevant documents from the entire collection is not sufficiently large, the queries are more difficult to answer. Bendersky and Croft (2009) showed that, in the case of longer queries, the users tend to click on lower-ranked results more often, implying that

DOI: 10.1002/asi

longer queries are relatively more difficult to answer than shorter queries. Kumaran and Carvalho (2009) proposed solving long queries by substituting them with shorter subqueries that will potentially lead to better result quality. In Balasubramanian et al. (2010), the long query reduction problem is addressed in the context of web search. Huston and Croft (2010) also focused on long queries and reported that simply reducing the length of a query by learning and removing stop structures can improve the retrieval performance. In Downey et al. (2007), characteristics of rare and popular queries were investigated. It was shown that, in the case of rare queries, the users click on fewer search results and perform a larger number of query reformulations, indicating that rare queries are more difficult to answer compared to popular queries. None of these works considered the number of retrieved results as a sign of query difficulty.

We note that this work is an extension of our previously published short paper (Altingovde et al., 2012), which included a subset of the contributions in the current paper. In this extended version, we provide a more detailed analysis of FNAQs and explore the impact of time on solving such queries. We also present a characterization of NAQs in terms of some quantitative features. Finally, the study on the feasibility of predicting NAQs is an entirely new contribution.

Research Objectives and Findings

The paper tries to answer the following research questions (RQs):

- Handling FNAQs
  - RQ1. What presentation mechanisms do commercial web search engines employ to provide the search results in the case of FNAQs?
  - RQ2. Does special handling of FNAQs help to increase the number of matching results?
  - RQ3. What modifications are observed in the queries suggested as an alternative to FNAQs?
  - RQ4. Are the queries suggested as an alternative to FNAQs relevant?
  - RQ5. Are more FNAQs solved by search engines over the course of time?
- Characterizing NAQs
  - RQ6. What are the root causes of NAQs?
  - RQ7. How do NAQs affect the user satisfaction with search results?
  - RQ8. What are the common features of NAQs?
- Predicting NAQs
  - RQ9. Can we predict, using machine-learning techniques, that a query is an NAQ before submitting the query to a search engine?

The main findings of our work can be summarized as follows:

- Handling FNAQs: Search engines present their results in four different ways (original query results, original query results + a suggested query, suggested query's results, no results). In all three search engines we examined, about one fourth of the FNAQs are answered by returning the results of a suggested query instead of the original user query. That is, the search engines have made an attempt to correct the original user query. According to our analyses, suggested queries tend to match more answers than the FNAQs. We also found that the presence of URIs in queries can make a big difference in the way suggested queries are generated for FNAQs. Through a user study, we showed that the query suggestions are perceived as useful by the endusers. Finally, we observed that the time has little impact on solving FNAQs.
- Characterizing NAQs: About one fifth of the NAQs do not contain any meaningful intent. Therefore, it may not be possible to solve them. Similarly, about one fourth of the NAQs include at least one nonexistent URI. Through a query log annotation study (i.e., by annotating the subsequent actions of every user who submitted an NAQ), we showed that, in the great majority of the cases, the users are dissatisfied by the NAQs. Finally, we found that NAQs usually contain spelling errors, are longer than common queries, and are more likely to contain terms that are not in the vocabulary of search engines.
- Predicting NAQs: We observed that frequency- and length-based features are the most useful features for predicting NAQs. Despite the heavy skew in class sizes, we could build a prediction model that can achieve good prediction quality: the area under the curve (AUC) is around 0.95 when all features are used in the model, and around 0.9 even when term frequency features are not available; which is more likely for the mobile- and meta-search scenarios targeted in this paper.

Sampling FNAQs

Our work requires using a sample set of real-life search queries for which a web search engine is likely to return very few or no answers. It is highly unlikely that any search engine company would make such a query sample public, as this would possibly expose confidential information about the techniques used (or not used) by the web search engine and even show its weaknesses. Therefore, in our work we rely on queries obtained from the AOL query log (Pass, Chowdhury, & Torgeson, 2006), which is the largest publicly available query log at the moment.

An exhaustive approach to sample a representative set of FNAQs from the AOL query log is to submit all unique queries in the log to different search engines and select queries that match few results. Unfortunately, this approach is not scalable due to the query limits imposed by search engines. Moreover, such an exhaustive sampling may be unnecessary, as most submitted queries would match a large number of results and not be interesting to our work. Therefore, we decided to make use of a query-result set which was previously obtained by issuing 660K unique AOL queries to the Yahoo! Web Search API in December 2010 (Altingovde, Ozcan, & Ulusoy, 2011). From this set, we selected queries with no matching results (around 16K

DOI: 10.1002/asi

TABLE 1. Number of queries that return k or fewer results.

k	A	B	C
0	244 (2%)	1,997 (17%)	1,791 (15%)
2	1,129 (10%)	6,377 (55%)	6,368 (55%)
10	3,829 (33%)	7,721 (66%)	7,366 (63%)
100	7,394 (63%)	9,089 (78%)	8,960 (77%)

Note. Throughout the paper, we arbitrarily name the three search engines A, B, and C.

queries) and resubmitted them to the same search API in July 2011. We observed that the number of queries without any matching results dropped to 11K queries (possibly due to updates in the API index, as discussed below). In the following two sections, we used these 11K queries as a representative set of queries that are likely to be FNAQs in practice.

In an earlier study, McCown and Nelson (2007) claimed that query results obtained from the API of a search engine may differ from those obtained from its web interface, as the index employed by the API is likely to be smaller than the full web index. Therefore, to verify that our final query sample is indeed made up of mostly FNAQs, we issued the 11K queries in our sample to three major search engines (Bing, Google, and Yahoo!) and obtained the first result pages (again, in July 2011). We note that, for all three search engines, we submitted the queries to the U.S. frontends,[1] which are supposed to have the largest index. We made sure to the greatest extent possible that all nondefault search preferences are disabled. We note that, while we conducted our experiments, Bing was providing the search results of Yahoo!. Nevertheless, we preferred to include both search engines in our experiments as i) even though the overlap observed between the results of the two search engines was not very low, the results were observed to be not identical, and ii) the two search engines seemed to employ different query correction mechanisms, leading to differences in results of certain queries.

In Table 1, for the three search engines, we report the number of queries that match k or fewer results. As we will discuss below, for some queries, search engines suggest another query and directly displays the results for this suggestion (while providing the option of seeing the results for the original query). In Table 1, we present only the number of results retrieved for the *original* query, but not for such a possible suggestion. According to the table, a large fraction of the 11K queries submitted to the three search engines return very few or no answers. For instance, search engines A, B, and C return fewer than 10 results for 33%, 66%, and 63% of queries, respectively. Moreover, 2% to 17% of these queries turn out to be actual NAQs. In the light of these numbers, we believe that we can safely refer to the queries in our sample as FNAQs. This means that FNAQs constitute around 1.6% (i.e., 11K/660K) of our log excerpt.

We note that, since the initial sampling of queries was performed using the Yahoo! Web Search API, we might have a slight bias towards those queries that could not be solved by Yahoo!. To investigate if such a bias exists, from the AOL query log we randomly sampled 6K singleton queries that were not in our initial 16K queries (nonsingleton queries are likely to be solved by all three search engines). We issued these queries to the three search engines and observed the matching result counts. This experiment showed that the ranking of search engines with respect to the percentage of NAQs is the same as in Table 1. However, as expected, the absolute numbers were much smaller. Hence, we believe that the way we sampled FNAQs does not introduce a significant bias towards any search engine.

Handling FNAQs

Our comparative analysis in this section aims to reveal how FNAQs are handled in web search engines. Obviously, like any other query, FNAQs are subject to many types of preprocessing, such as spelling correction, query rewriting, or term expansion. A search engine may apply to all or a subset of these to improve the result quality. Unfortunately, most of this processing happens at the backend search system and the details are not visible to us. Therefore, herein, we opt for an alternative study: We analyze the search result pages retrieved as response to FNAQs and observe what kind of manipulation is performed over the original user query and how the results are presented. This analysis gives us a hint about the actions taken by search engines when handling FNAQs.

RQ1. What Presentation Mechanisms Do Commercial Web Search Engines Employ to Provide the Search Results in the Case of FNAQs?

We start with analyzing the result presentation patterns of search engines A, B, and C in response to queries. In all three search engines, we observe four types of patterns:

- *Original Query Results (OrgRes)*: The results of the original user query are returned to the user without any explicit modification on the original query.
- *Original Query Results With a Suggested Query (OrgResSugQuery)*: When the results of the original query are retrieved, also a new query is suggested. In this case, the search engine believes that the original query results are good enough, but it gives the user the option of checking the results of another query.
- *Suggested Query Results (SugQueryRes)*: The user is provided with the results of another query instead of the original query. In this case, the search engine believes that the real user intent matches another query, which potentially retrieves better or more results than the original query.
- *No Results (NoRes)*: No matching results are returned. In this case, the search engine also fails to suggest a related query.

[1]We ensured this by issuing the queries to the main search frontends (i.e., those without any region extension) for Google and Yahoo!, and by selecting the U.S. region (English) for Bing (as it yields fewer NAQs than the international option).

DOI: 10.1002/asi

TABLE 2. Result presentation patterns in different search engines.

Pattern	SE	Message displayed in the search engine result page.
OrgRes	All	—
OrgResSugQuery	Bing	Do you mean <suggested query>.
	Google	Did you mean: <suggested query>.
	Yahoo!	Did you mean: <suggested query>.
SugQueryRes	Bing	No results found for <original query>. Showing results for <suggested query>.
	Google	Showing results for <suggested query>. Search instead for <original query>.
	Yahoo!	We have included <suggested query> results. Show only <original query>.
NoRes	Bing	No results found for <original query>.
	Google	Your search – <original query> – did not match any documents.
	Yahoo!	We did not find results for: <original query>.

Each of the three search engines we analyzed has its own way of presenting the results. The result presentation patterns we observed are summarized in Table 2. In Table 3, for the three search engines, we report the number of query results falling under each pattern. According to the table, all three search engines make a considerable effort to correct the original query by providing query suggestions (via pattern *OrgResSugQuery*) or, more directly, by providing the suggested queries' results (via pattern *SugQueryRes*). Search engine A prefers to provide the original results (pattern *OrgRes*) for the majority of queries (around 62%), whereas B and C handle such queries mostly via the *OrgResSugQuery* pattern. Also, a quick comparison between Tables 1 and 3 shows that the *SugQueryRes* pattern seems to help B and C to reduce the percentage of NAQs (*NoRes* pattern) substantially. Obviously, the quality of the suggested query and corresponding results should also be investigated to justify the benefits of the *SugQueryRes* pattern other than transforming an NAQ to a query with answers. We address the former issue while answering RQ4 later, and show that suggested queries are mostly found relevant to the original information request.

RQ2. Does Special Handling of FNAQs Help to Increase the Number of Matching Results?

The results above show that the percentage of NAQs can be reduced by generating a new, potentially related query, and presenting the results of this query (i.e., *SugQueryRes* pattern), instead of the original query. However, it is also important to check whether these new queries lead to an increase in the number of retrieved results. Table 4 reports the number of queries that return k or fewer results (the column for *NoRes* pattern is repeated from Table 3 for the sake of completeness). We observe that there are more matching results when the results of a new query are returned (*SugQueryRes* pattern) compared to the original query results (patterns *OrgRes* and *OrgResSugQuery*). For instance, C returns fewer than 10 results for 71% and 49% of queries when patterns *OrgRes* and *OrgResSugQuery* are observed, respectively, but only 5% of queries have fewer than 10 results when *SugQueryRes* pattern is observed.

TABLE 3. Number of queries with a certain pattern.

SE	*OrgRes*	*OrgResSugQuery*	*SugQueryRes*	*NoRes*
A	7,267 (62%)	1,277 (11%)	2,896 (25%)	233 (2%)
B	3,519 (30%)	4,584 (39%)	3,068 (26%)	502 (4%)
C	2,771 (24%)	5,340 (46%)	3,101 (27%)	461 (4%)

A clearer picture of the impact of the *SugQueryRes* pattern on the number of results can be seen by comparing the "Total" column of Table 4 to Table 1, for each search engine and k value. For instance, Table 1 shows that search engine C would retrieve at most two results for 6,368 queries (Recall that what Table 1 reports is based on the number of results for the original query, but not the suggested query via some pattern). By presenting the results via the *SugQueryRes* pattern for some of the queries, the number of queries with at most two results drops to 3,631 for C. Similar reductions are observed for other cases as well.

These findings imply that the number of matching results can be increased by proper handling of FNAQs (e.g., showing the results of another, related query). Note that, as mentioned before, the quality of the results retrieved by these patterns is an open question, and as a step towards answering the latter question, we evaluate and verify the quality of the suggested queries in the upcoming section.

RQ3. What Modifications Are Observed in the Queries Suggested as Alternative to FNAQs?

In the case of *OrgResSugQuery* and *SugQueryRes* patterns, the search engine suggests an alternative query that is usually obtained by modifying the original query. To have a clue about the underlying strategies that generate these suggested queries, we manually inspected all queries that led to patterns *OrgResSugQuery* or *SugQueryRes* in all three search engines (i.e., a total of 1,459 queries). In Table 5, we present the modifications commonly encountered during this manual inspection.

DOI: 10.1002/asi

TABLE 4. Number of queries that return k or fewer results.

SE	k	*OrgRes*	*OrgResSugQuery*	*SugQueryRes*	*NoRes*	Total
A	2	544 (8%)	126 (10%)	4 (0%)	233 (100%)	907
	10	1,602 (22%)	435 (34%)	17 (1%)	233 (100%)	2,287
	1000	4,530 (62%)	751 (59%)	196 (7%)	233 (100%)	5,710
B	2	2,040 (58%)	1,295 (28%)	96 (3%)	502 (100%)	3,933
	10	2,785 (79%)	2,094 (46%)	189 (6%)	502 (100%)	5,570
	1000	3,059 (87%)	3,099 (68%)	621 (20%)	502 (100%)	7,281
C	2	1,496 (53%)	1,605 (30%)	69 (2%)	461 (100%)	3,631
	10	1,965 (71%)	2,608 (49%)	157 (5%)	461 (100%)	5,191
	1000	2,233 (81%)	3,747 (70%)	557 (18%)	461 (100%)	6,998

TABLE 5. Frequently encountered modifications in the case of queries without a URI (M1–M5) and with a URI (M6–M11).

Modification	Original query	Suggested query
M1 Split query string to terms	3rdgenerationgospelsingers	3rd generation gospel singers
M2 Correct typo in a term	tadeair compter show	t**r**ade air comp**u**ter show
M3 Combine terms in query string	cup cakes	cupcakes
M4 Add/delete punctuation	childerns hosptial of birmaham	child**re**n**'**s hosp**it**al of birm**ing**ham
M5 Add/delete/replace term	woodiestationwagons	**woody** station wagons
M6 Split URI to terms	www.eldercare-today.com	elder care today
M7 Correct typo in a term in URI	www.orlandocollages.com	www.orlandocolleges.com
M8 Add/delete/replace term in URI	www.online-houses-for-sale.com	www.online-homes-for-sale.com
M9 Re-order terms in URI	ri-rvs.com	**rvs-ri**.com
M10 Add/delete punctuation	street-racingvideos.com	street-racing**-**videos.com
M11 Add/delete/replace domain	learndirect-advice.co	www.learndirect-advice.co.uk
	www.innuendo-music.de	www.innuendo-music.com

During the manual inspection, we found that a large number of queries entirely or partially include a URI. Indeed, among the 273 queries that fall under the pattern *OrgResSugQuery*, the percentage of queries with a URI was found to be 71%. For the pattern *SugQueryRes*, the percentage is 52%, which is lower but still significant. As shown in Table 5, the modifications observed in queries differ depending on the presence of a URI in the query. In particular, for queries without a URI, the most common modifications are adding a space between the words and then correcting the typos within the terms. On the other hand, 70% of URI-bearing queries do not involve any obvious typo. But, there are mistakes possibly due to the poor memory of users who typed a wrong domain (e.g., typing ".co" instead of ".co.uk") or forgot the hyphen between the terms (see the examples in Table 5).

Finally, note that an observed modification can be obtained via a particular technique or a combination of several techniques, and conversely, several modifications that appear to be different can indeed be produced by a single technique. For instance, one may compute the edit-distance of an entire query string (or its terms) to the existing queries (or terms) as well as the distance of a URL to the known URLs in the search engine's database to obtain modifications as M2 and M7, respectively, in Table 5. While we present the final effect on the suggested queries, we do not discuss the methods that generate them, as these methods are not made public by search companies.

RQ4. Are the Queries Suggested as an Alternative to FNAQs Relevant?

As a complementary experiment, we investigated the quality of the suggested queries. To this end, we randomly selected two subsets, each with 100 queries, from the queries that led to patterns *OrgResSugQuery* or *SugQueryRes* in all three search engines. We then conducted a user study with six judges, that is, some of the coauthors of this work as well as other graduate students in the field of computer science. Each judge was shown the original query and the suggested query obtained from each search engine, and was asked to decide if the suggested query makes sense or not. The tasks were exclusive, that is, every query and corresponding suggestions are annotated by a single judge.

In Figure 1, we show the percentage of query suggestions labeled as relevant, irrelevant, and undecided by the judges. The figure shows that search engine A has the lowest number of irrelevant results in the case of the pattern *OrgResSugQuery*. However, a large number of query suggestions are labeled as undecided for this search engine. A closer inspection reveals that A consistently prefers to provide alternative URI suggestions, whereas the other two search engines prefer to split the URI into multiple terms. This choice of A yields lots of undecided suggestions, as the judges could not decide on how good the suggested URI captures the initial intent of the user in a number of cases.

In the case of pattern *SugQueryRes*, all search engines provide a larger percentage of relevant suggestions in

DOI: 10.1002/asi

comparison to pattern *OrgResSugQuery*. This result further confirms our intuition that search engines trigger the former pattern *SugQueryRes* only when they are confident about their suggestion.

RQ5. Are More FNAQs Solved by Search Engines Over the Course of Time?

In order to investigate the impact of time on our findings, we repeated the previously reported experiments at a later date, in January 2012 (the previous experiments were conducted in July 2011), considering only search engine A, which achieves relatively low FNAQ ratios. For brevity, we limit the discussion to the experiment reported in Table 3. Table 6 shows the number of queries whose patterns had changed between July 2011 and January 2012. According to the table, the percentage of NAQs decreases over the time, that is, there are fewer queries under the *NoRes* pattern. In particular, 142 queries that were under *NoRes* pattern in July 2011 had moved under the *OrgRes* pattern by January 2012. On the other hand, we observed that 127 of those 142 queries still matched fewer than 10 results, that is, they are still FNAQs.

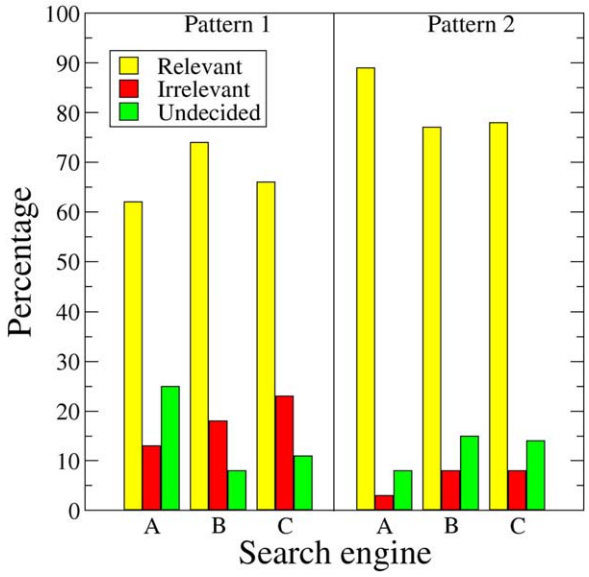

FIG. 1. Suggestion quality of search engines. [Color figure can be viewed at wileyonlinelibrary.com]

Table 7 reports the number of queries that returned k or fewer results in January 2012. A quick comparison of Table 7 and corresponding Table 4 reveals that for each search engine, the number of queries with fewer than 1,000 results drop, which indicates that for some of the FNAQs there are more retrieved results now. The highest improvement is for engine C, for which the number of queries with less than 1,000 answers drop from 6,998 to 5,943 in January 2012, and this means that for only 9% of the queries the situation becomes better (possibly due to newly crawled pages and/or search engine's ability to apply a certain pattern [like the patterns *OrgResSugQuery* or *SugQueryRes]* based on the new evidence, such as the submission of queries that are similar to a particular FNAQ); while a considerable number of queries (i.e., more than 30% of our query set) still retrieve fewer than 10 results. Furthermore, the number of NAQs increases for search engines B and C (but not A, as discussed in the previous paragraph).

These observations justify our motivation to investigate FNAQs. We find that such queries are difficult to be resolved in time by external factors (e.g., the growth of the web and creation of webpages that may match such queries). They rather call for special treatment, which may also

TABLE 6. The temporal change in the patterns triggered by search engine A.

	January 2012				
July 2011	*OrgRes*	*OrgResSugQuery*	*SugQueryRes*	*NoRes*	Total
OrgRes	6,786 (93%)	270 (4%)	172 (2%)	39 (1%)	7,267
OrgResSugQuery	172 (13%)	856 (67%)	244 (19%)	5 (0%)	1,277
SugQueryRes	136 (5%)	366 (13%)	2,393 (83%)	1 (0%)	2,896
NoRes	142 (61%)	29 (12%)	4 (2%)	58 (25%)	233
Total	7,236	1,521	2,813	103	

TABLE 7. Number of queries that return k or fewer results in January 2012.

SE	k	*OrgRes*	*OrgResSugQuery*	*SugQueryRes*	*NoRes*	Total
A	2	296 (4%)	46 (3%)	1 (0%)	103 (100%)	446
	10	3,012 (42%)	526 (35%)	13 (1%)	103 (100%)	3,654
	1000	4,414 (61%)	865 (57%)	154 (6%)	103 (100%)	5,536
B	2	1,132 (41%)	1,636 (43%)	67 (2%)	782 (100%)	3,617
	10	1,464 (54%)	2,236 (58%)	206 (5%)	782 (100%)	4,688
	1000	1,900 (69%)	2,953 (77%)	644 (15%)	782 (100%)	6,279
C	2	739 (30%)	1,875 (43%)	65 (1%)	521 (100%)	3,200
	10	1,019 (42%)	2,517 (61%)	219 (5%)	521 (100%)	4,276
	1000	1,494 (62%)	3,238 (79%)	690 (15%)	521 (100%)	5,943

DOI: 10.1002/asi

TABLE 8. Example NAQs that are manually selected from the AOL query log.

NAQ	Potential reason for not matching any result
iiugjjgjvjgkygjfgkghjdjkskjfsdhfdhgfyugdf	Query term does not appear in the vocabulary
elszabe tcollage	Query has typos that cannot be fixed by the spelling corrector
talkshowswww.mauray.com	URI does not exist in the web
healperware tea & coffee pot made in china	No webpage contains all of the query terms
;;;;;;;;;;;;;;;;	Query has insufficient information
www.picgallery.katrinaloca04.org	URI is not discovered by the search engine

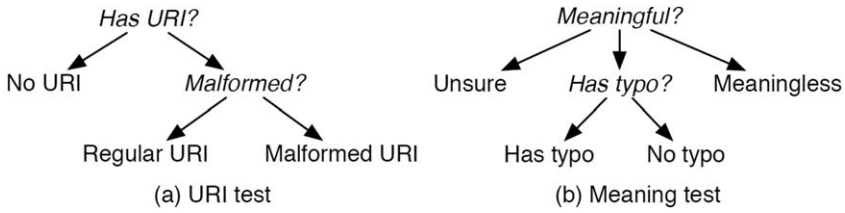

FIG. 2. Procedure followed by the judges in the study.

improve the result quality and, potentially, the user satisfaction with search results.

Characterizing NAQs

In this section we focus on queries that match no answers (NAQs), which is a very specific and more problematic subset of FNAQs. In our analyses, we use queries that match no results in at least one of the three search engines, which add up to 665 queries in total. Table 8 shows a small number of NAQs selected from the mentioned set, together with the potential reason for being an NAQ.

RQ6. What Are the Root Causes of NAQs?

To identify the root causes of NAQs, we conducted a user study. NAQs are labeled by four judges (who are among the authors) based on two types of tests: URI presence and meaningfulness. As these tests are not complicated and the judges generally agreed on each other's labels for a small number of queries, we assigned each query to only one judge, exclusively. Figure 2 illustrates the query labeling procedure followed by the judges. We report the results, separately, for each search engine, as well as the union and intersection of their NAQ sets.

Our first test evaluates the presence of URIs in NAQs. Although it could be possible to automate this test via pattern-matching techniques, we prefer to do it manually, as it is difficult to automatically catch URIs that contain typos. The results in Figure 3 indicate that about 57% of the NAQs contain at least one URI. About 45% of those contain at least one malformed URI, while the remaining 55% are proper URIs (i.e., conforming to the syntax as described by the RFC 3986 URI Generic Syntax). This shows that about one fourth of NAQs aim to retrieve resources that are unknown to or not discoverable by the search engine. Hence, it is

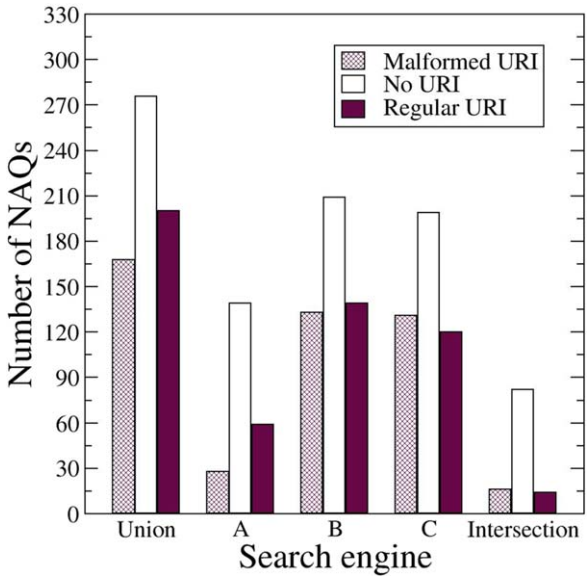

FIG. 3. Distribution of NAQs based on URI presence. [Color figure can be viewed at wileyonlinelibrary.com]

highly unlikely that these NAQs can be solved by a query-handling technique. When we compare the results across the three search engines, we observe that search engine A is significantly better in solving NAQs with malformed URIs. The number of such NAQs in A is only slightly higher than those present in the intersection set of the three search engines. Overall, the size of the intersection is much smaller than the size of the union, which implies that most NAQs with a URI are solved by at least one search engine.

Our second test is about the meaningfulness of NAQs. If a query contains a URI, we only consider the remaining query terms. If the entire query is a URI, it is labeled as "only URI" and excluded from the test (we found that 323

DOI: 10.1002/asi

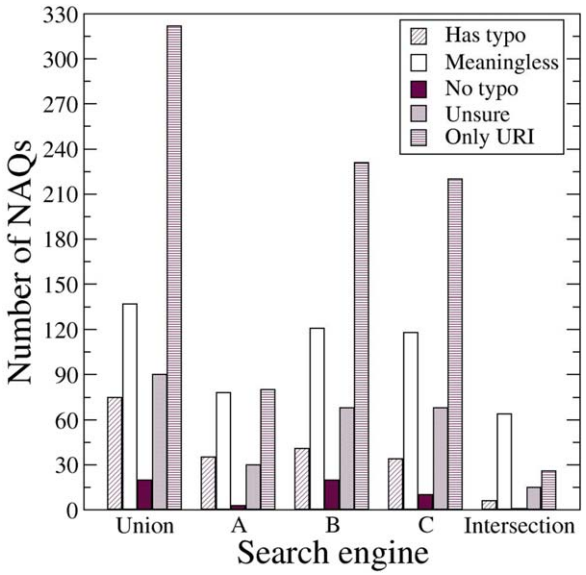

FIG. 4. Distribution of NAQs based on meaningfulness. [Color figure can be viewed at wileyonlinelibrary.com]

queries out of 665 fall into this case). If the meaning of an NAQ is not clear to the judge, but the NAQ has a potential to have a meaning for the user who issued it, then the judge labels the NAQ as "unsure." NAQs that are clearly meaningless to the judge (e.g., queries that are only formed of repetitive key strokes) are labeled as "meaningless." The remaining NAQs are considered "meaningful" and labeled as "has typo" or "no typo," depending on the presence of a typo.

The results of this test are shown in Figure 4. Since a considerable portion of the NAQs are labeled as "unsure" (i.e., 89 out of 665), the numbers reported for the remaining labels can act only as lower bounds. We found that at least 137 and 116 queries (out of 665) can be considered as meaningless and meaningful, respectively. According to the results, only 3% of NAQs (21 out of 116 queries) are meaningful and do not contain any typos. It is interesting to note that, in our study, we encountered only one such NAQ that is not solved by either search engine. At least four out of every five NAQs that are meaningful contain a typo (i.e., 95 queries out of 116) that is not fixed by the spell-checker. At least 21% of NAQs do not have any meaning. This final result sets an upper bound of 79% on the fraction of NAQs (i.e., 525 queries) that a search engine can fix by employing more sophisticated techniques.

RQ7. How Do NAQs Affect User Satisfaction With the Search Results?

Next, in order to provide some insight into potential user dissatisfaction with NAQs, we performed an editorial study involving annotations. We sorted the AOL query log (Pass et al., 2006) first with respect to the user-id and then time. Then we identified every NAQ in the log, and manually annotated what the user has done after submitting this NAQ.

TABLE 9. User experience after submitting an NAQ.

User experience	Pattern	Share (%)
Dissatisfied	TS	46.2
	NQ	19.6
	RQ-noClick	22.7
Satisfied	RQ-click	11.5

We classified the user behavior into one of the following three cases:

- *User terminates the session* (TS): After submitting the NAQ, the user has no further activity in the current search session. Following the practice in the literature, we assumed that the session timeout period is 30 minutes, that is, if an NAQ is not followed by any action in the next 30 minutes, the current search session is assumed to end.
- *User submits a new query* (NQ): The user submits a new query with no explicit syntactic or semantic similarity to the previous query (i.e., the NAQ) in the current session.
- *User submits a reformulated query* (RQ): The user submits a new query that modifies an NAQ in the current session and the new query has some syntactical or semantic similarity to the NAQ. We note that there are two further possibilities in this case: The user can submit a chain of reformulations without clicking on the retrieved results (if any) and finally abandons his or her search by falling into one of the TS or NQ cases described above. Alternatively, after submitting one or more reformulated queries, the user clicks on at least one of the search results. We label these two subcases as RQ-noClick and RQ-click, respectively.

Based on this classification, we dub a search session dissatisfactory if an NAQ is followed by one of the TS, NQ, or RQ-noClick cases, as the user abandons her or his search (either directly or after a series of query reformulations) without clicking any results. In case of RQ-click, it is not clear whether the user could really find what s/he was looking for. But even if this is the case, the user can satisfy her or his information need after some effort, through an explicit reformulation of the query (possibly more than once). Nevertheless, to be on the safe side, we do not consider this latter case as a dissatisfactory experience. In Table 9, we provide the results of our study for 665 NAQs. NAQs that fall into the TS or NQ cases account for 65.8% of all NAQs. Furthermore, 22.7% of NAQs fall into the RQ-noClick case. Therefore, we can safely claim that in at least 88.5% of cases the user was dissatisfied.

RQ8. What Are the Common Features of NAQs?

In Figure 5 we display the distribution of NAQs and common queries (an equal number of queries that are again sampled from AOL log and match at least one result) as the query length increases (computed by using a homemade query parsing tool). We observe that a significant fraction of NAQs do not contain any query terms after normalization (e.g., removal of punctuation), while almost every common

DOI: 10.1002/asi

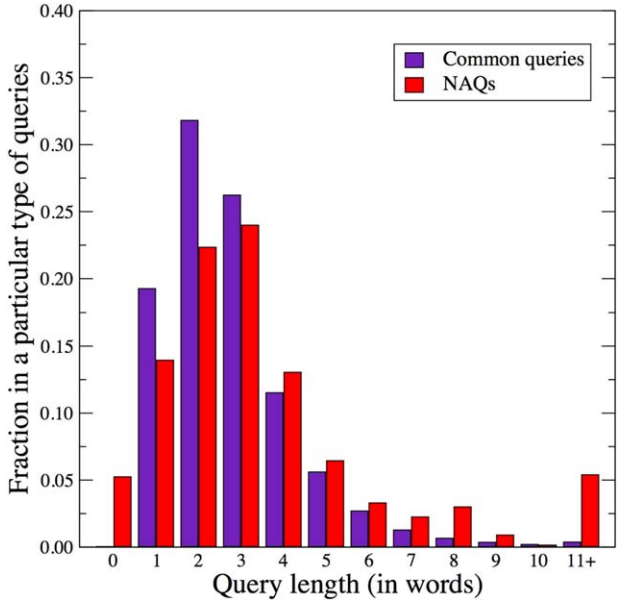

FIG. 5. Query length (in words) distribution. [Color figure can be viewed at wileyonlinelibrary.com]

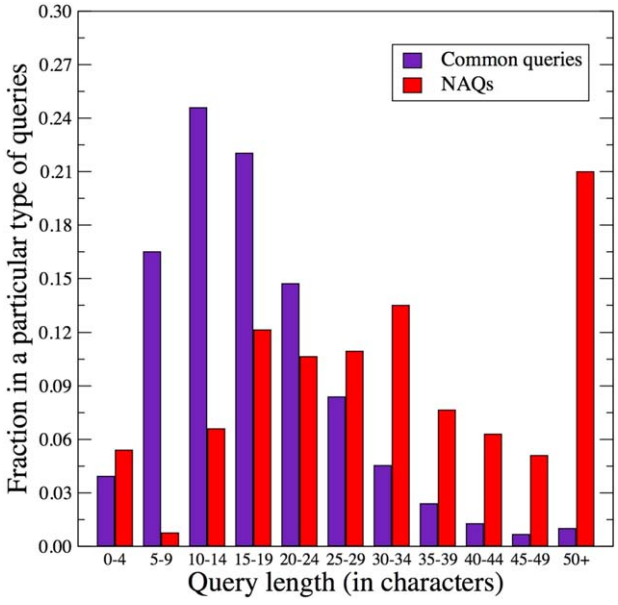

FIG. 6. Query length (in characters) distribution. [Color figure can be viewed at wileyonlinelibrary.com]

query contains at least one term (there were only two common queries with no terms). The fraction of NAQs for queries with one to three terms is lower than those for common queries. The NAQ distribution is slightly shifted towards longer queries. Overall, this behavior can be explained by two observations. First, it is well known that most web queries include one to three terms. Thus, NAQs are not likely to dominate this range. Second, as there are more terms, it becomes harder to match the query to a document that contains all query terms. The second factor becomes dominant at large query lengths. Figure 6 shows the behavior in terms of the number of characters in the query. We observe that the NAQ likelihood is more skewed towards queries with many characters, compared to common queries.

We also investigated the relationship between the terms in NAQs and their document frequency. To this end, we issued all such terms (1,770 terms that are extracted from NAQs after usual normalization like removing white space and punctuation) to all three search engine frontends and retrieved the matching result counts. Figure 7 shows the fraction of NAQ terms for a certain number of matching results. We observe that a large fraction of the terms in NAQs are not in the vocabulary of search engines (5% to 8%).

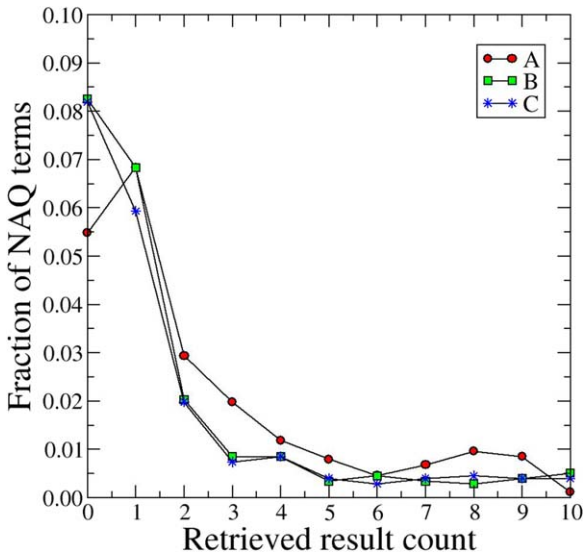

FIG. 7. Number of NAQ terms with a certain document frequency. [Color figure can be viewed at wileyonlinelibrary.com]

Predicting NAQs

In this section, we investigate the feasibility of predicting NAQs. Predicting whether a query is an NAQ before it is submitted to the search engine can be useful in several scenarios:

- *Mobile search*: A predictive model deployed within a mobile device can warn the user if the query is not likely to return any answers, providing savings in terms of time, bandwidth usage, power consumption, and monetary costs.
- *Meta-/federated search*: The meta-search system (or the broker) can build a separate NAQ predictor for each search service and can forward the query to only those services that are predicted to return some results. This may reduce the bandwidth usage and financial costs of the broker while reducing the load on the search services.

RQ9. Can We Predict, Using Machine-Learning Techniques, That a Query Is an NAQ Before Processing the Query in a Search Engine?

We cast the problem of predicting whether a query will return no results as a classification problem and solve it

DOI: 10.1002/asi

TABLE 10. The features used in the prediction model.

Type	Feature	GE	AV	AC
User	loggedIn	√	√	√
	country	√	√	√
Temporal	hourOfDay	√	√	√
Length	#OfWords-X		√	√
	#OfChars-X		√	√
	avgWordLength-X		√	√
Frequency	queryFreq-X			√
	totalWordFreq-X			√
	avgWordFreq-X			√
	minWordFreq-X			√
Others	#OfPlusSymbols-X		√	√
	#OfMinusSymbols-X		√	√
	#OfQuotationSymbols-X		√	√
	fracOfURLs-X		√	√
	#OfDigits-X		√	√
	#OfUpperCases-X		√	√
	#OfAlphaNumeric-X		√	√
	fracOfDigits-X		√	√
	fracOfUpperCases-X		√	√
	fracOfAlphaNumeric-X		√	√

using machine-learning techniques, that is, we have a binary classification problem where query instances belong to the "no answer" or "common" categories.

The features used by the learned model are given in Table 10. Our choice of features is guided by the characteristics of the NAQs presented in the previous section as well as the earlier works in the literature. In particular, due to the similarity of our problem to the problems of difficult/rare query prediction and query performance prediction, our feature set essentially includes the core features employed in the previous works addressing these problems. One such feature is the query length, which can be computed in various ways (e.g., in terms of the number of words or characters (Balasubramanian, 2010; He & Ounis, 2006). The query length is reported to be different for popular and tail queries (as the latter type of queries are usually longer) and our NAQs being quite rare queries at the tail, this feature may be discriminative for predicting them. Another group of features is based on the collection-frequency of query terms. These features are motivated by the intuition that rare terms are less likely to appear together, and hence they are more likely to yield no results. Earlier work on query quality prediction also employ features of the same flavor, such as minimum and maximum of the inverse document frequency (IDF) scores of query terms (Kumaran & Carvalho, 2009) or their standard deviation (He & Ounis, 2006). We also construct several lexical features (e.g., presence of symbols, digits, or URIs in the query string), which were employed before in the query performance prediction task (Balasubramanian et al., 2010). The country of the user could be a useful feature because the web coverage of a search engine may be limited for certain countries or high-quality natural language processing (NLP) tools may not be available to handle queries issued from those countries, leading to NAQs in both cases. In some search engines, the query results are personalized if the user is logged into the system, and this may affect the retrieved results. Therefore, we use the information about whether the user is logged into the system or not as a separate feature.

We compute certain features (those whose names end with an X) for three different versions of the query string: original, spelling-corrected, and normalized. The spelling-corrected query (if available) is provided by the search engine. We obtain the normalized query ourselves. We form three sets of features based on different criteria, considering the fact that not all features may be available in all use case scenarios. For instance, while all query- and term-related features may be available to a search engine, a mobile application may not be able to access these features to predict NAQs. More specifically, we investigate the prediction accuracy of subsets of features, taking into account the generalizability and availability of our features. The GE (generalizability) set contains the features that are available independently of a query log and a document collection. These features allow learning models, independent of a particular search engine. The AV (availability) set contains the features that are likely to be available to a search client. In our case, frequency-related features that require storing large amounts of frequency data on the client side are removed from the full feature set. Finally, the AC (accuracy) set contains all features. We train a separate classifier for each one of these three feature sets.

We sampled queries from two consecutive days of Yahoo! Web Search query logs (about 16 million queries in total). The queries were normalized by lowercasing every query term, removing stop-words and duplicate query terms, ignoring query terms longer than 20 characters, and removing queries whose language has no space separator. For training, we used queries from the first day and, for testing, we used queries from the second day. To prevent the class imbalance in the training set, we downsample common queries such that the training set contains a similar number of NAQs and common queries. While testing the model, we used the original distribution in the test set.

As the classification technique, we used gradient boosted decision trees (GBDTs) (Friedman, 2000). An advantage of the GBDT classifier over other learners is that its output is easy to interpret, since it includes an explanation of the relative influence of different variables. As another advantage, the GBDT classifier has been shown, many times, to achieve high classification accuracy values and outperform other machine-learning techniques in various classification tasks (Zheng et al., 2007).

Due to the high class imbalance, we report the performance using the receiver operating characteristic (ROC) curve. The ROC curve plots the true-positive rate versus the false-positive rate. We also report the AUC as a summary of the performance of the classifier. The parameters ruling the behavior of the learning model (the shrinkage parameter, the number of trees, and the number of nodes per tree) are selected using a held-out query set.

DOI: 10.1002/asi

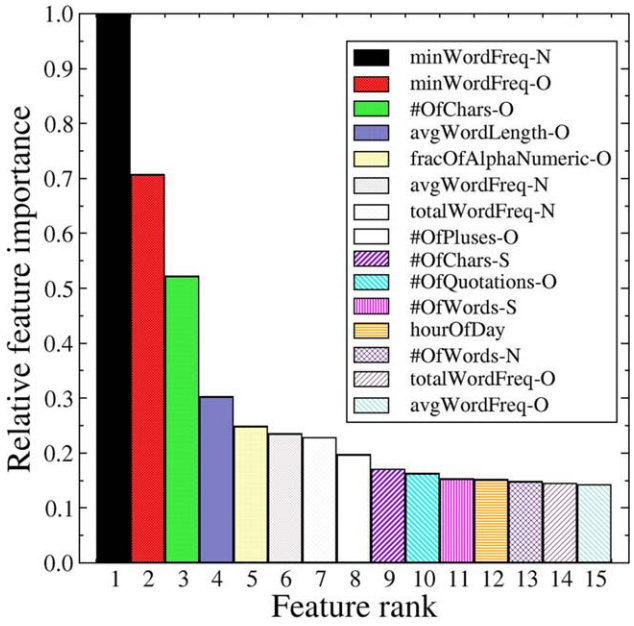

FIG. 8. Relative importance of the features. [Color figure can be viewed at wileyonlinelibrary.com]

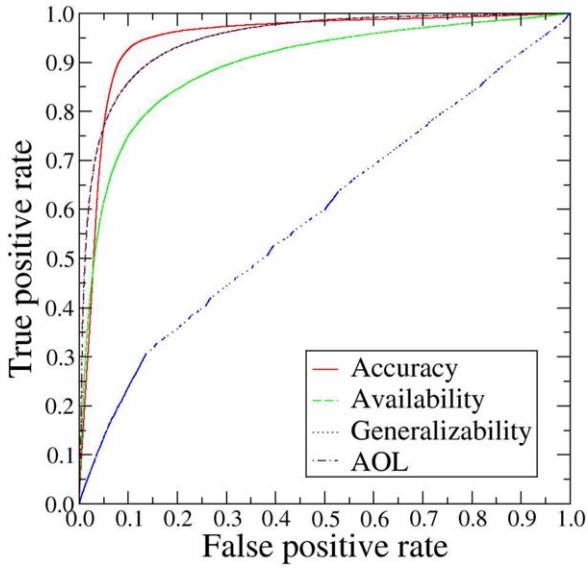

FIG. 9. ROC curves for the feature sets in Table 10. [Color figure can be viewed at wileyonlinelibrary.com]

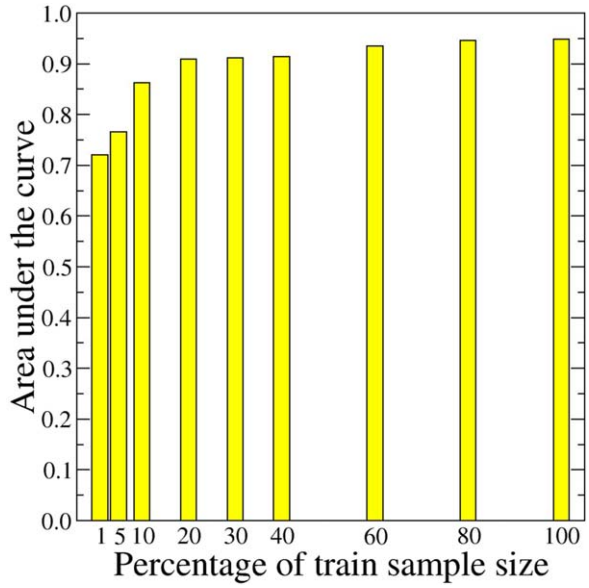

FIG. 10. Increase in AUC as more training data are used. [Color figure can be viewed at wileyonlinelibrary.com]

As the first step, we investigate the most influential features for classifying NAQs. The GBDT classifier provides a simple and intuitive way to interpret the relative importance of features. Figure 8 shows the normalized relative importance of the most influential 15 features. The features whose name ends with -O correspond to the features extracted from the unmodified original keywords, -N to their normalized version after processing the query, and -S to features extracted after spelling correction. We observe that the feature importance values are highly skewed, which implies that there are only a few powerful features. We also observe that frequency-based and length-based features are likely to have relatively high impact on the prediction performance. Frequency-based features that are computed using the normalized version of queries are more helpful than those computed using the original queries.

We evaluated the performance of the previously mentioned feature sets (GE, AV, and AC). The results are summarized in Figure 9, which shows a separate ROC curve for each feature set. In general, the features in the GE set perform close to a random assignment of classes (AUC of 0.591). In turn, the classifier that uses the features in the AV set is able to produce very good classification results (AUC of 0.894). This is important, as the latter set contains only those features that are likely to be available to a search client in our mobile- and meta- search scenarios. Using all features (the AC set) yields an even higher AUC of 0.949. This means that if term frequencies can be made available (say, by storing them at the client device together with the prediction model), the classifier is able to make very accurate predictions. This is a positive finding, given that the distribution of NAQs in the test set is highly skewed, and it implies that the features extracted are useful for classifying NAQs. In the same figure, we also display the performance of our model over the test queries selected from the AOL log (these are the queries we used before for characterizing NAQs). This verification is important to see if our model generalizes to different query logs. Indeed, we observe that the performance is comparable to the performance obtained by using the features in the AC feature set.

As another experiment, we varied the amount of training data fed into the classifier to determine the number of training instances that the classifier needs to achieve a good performance. We randomly split the initial training set obtained

DOI: 10.1002/asi

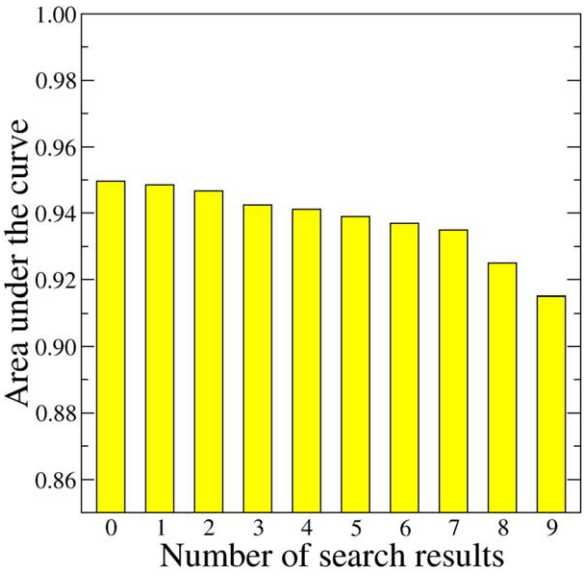

FIG. 11. Decrease in AUC as more results are returned. [Color figure can be viewed at wileyonlinelibrary.com]

from a single day into several subsamples containing a fraction of the training instances. The testing was performed using the same query log in all cases. Figure 10 presents the results, where the y-axis shows the AUC and the x-axis shows the percentage of training instances. In the figure, we can observe that the classifier is able to correctly predict 90% of the cases using only 20% of the original training data. This implies that the classifier needs only a small number of instances to perform well.

We also performed a set of experiments in order to check if the performance drops with time. Specifically, we trained the model at an origin day X and classified queries using a query log from day $X + Y$, where $Y \in \{2, 4, 8, 16, 32\}$. The model was trained using the full training query set coming from the first day X and tested using 8 million queries for each of the subsequent days. In all cases, the AUC and precision metrics did not show any significant difference with respect to those in the first day, with a variation in AUC less than 1% in all cases. This means that the classifier is very stable over time, a potentially important observation for the mobile search scenario. In particular, once a prediction model is downloaded (possibly through a wired connection) and deployed at a mobile device, it can be safely used for more than a month without requiring an update. Given that there is no significant difference in the performance across the data sets, we prefer not to demonstrate the results.

Finally, we assessed the performance of the classifier in predicting queries that match fewer than X results ($X \in \{0, \ldots, 9\}$). This assessment is important because such queries are potentially difficult and the search engine might want to be informed so that it can proactively suggest a reformulation of the query to the user. We generated 10 different pairs of training and test sets of sizes comparable to those used before and calculate the AUC as before. In Figure 11, we report the performance for every pair. The performance was observed to be stable across all 10 sets, dropping slightly when we increased the threshold for the number of results, with a maximum drop of 4% when $X = 9$. In general, the classifier predicts correctly, on average, more than 9 out of 10 queries. This result is in line with the results previously reported in this section.

Discussion on Solving NAQs

In this section we discuss some techniques that might be appropriate for handling NAQs exhibiting certain characteristics as described in previous sections.

NAQs with more than one term: The de-facto search semantics of real-life search engines is conjunctive, that is, retrieved results contain all query terms. In the case of NAQs, this can be relaxed for the sake of obtaining at least an approximate answer. A step towards this goal may be dropping some of the query terms and processing the remaining terms, again, in conjunctive mode. In this case, dropping terms with the minimum document frequency may be useful (dropping terms that have zero frequency is definitely useful) although this may overgeneralize the query. On the other hand, keeping terms with low frequency in the query may yield further NAQs and increase the response time if many subqueries need to be executed before an approximate answer is generated. An earlier study used machine-learning techniques to rank the subqueries of a long query and then replaces the original query with the best subquery to improve the retrieval quality (Kumaran & Carvalho, 2009). In the case of NAQs, the best subquery identified in this way may still return no answers, implying that the learning process should also take into account the need to generate an answer. As a more efficient alternative, the subsets of the query that are already stored in the search engine result cache (i.e., with known answers) can be used to obtain an aggregate answer (Cambazoglu, Altingovde, Ozcan, & Ulusoy, 2012).

A more liberal approach may be to switch to disjunctive matching semantics and rank documents that include any of the query terms. This approach may again suffer from overgeneralizing the query and/or high processing costs.

Finally, another possible solution is to adopt the methods that are proposed for generating query reformulations or suggestions (e.g., Jansen, Booth, & Spink, 2009; Jones, Rey, Madani, & Greiner, 2006). For instance, in the context of sponsored search, Jones et al. (2006) generate substitutions for terms, phrases, or entire queries using external semantic resources or other queries. This approach may not be appropriate for NAQs since many NAQs include at least one term with very low document frequency, and it is very unlikely that such terms can be found in other queries or semantic

DOI: 10.1002/asi

resources. Yet this might be a good approach to handle NAQs that stem from linguistic problems (e.g., when an awkward translation of a term from another language is used in the query).

NAQs with a single term: The solutions that drop terms are not viable for single-term NAQs. It is also difficult to make query suggestions or replacements based on queries with similar terms. Spelling correction may be a viable choice for such queries. However, given that these queries could not be answered at the moment, this may be considered an indication of the inability of simple spelling checkers to correct such typos. This calls for more advanced string processing techniques, such as first segmenting the word into many components (Pu & Yu, 2008) and then applying spelling correction.

While such techniques can yield some results for NAQs, whether these results would satisfy the user's information need is a different issue that has to be explored on its own. Further note that NAQs still exist, although the commercial search engines may have already adopted such mechanisms. This means that either new or more advanced mechanisms should be developed, or the existing ones should be applied more aggressively (possibly in combination). Since both options potentially increase the risk of hurting the performance for non-NAQ queries, we believe that such mechanisms and/or their aggressive use should be utilized only when a query is found or predicted to be an NAQ, but certainly not for all queries.

Conclusion

We provided a characterization of queries that match very few or no results in major search engines. After a detailed analysis on how queries with few results are handled by search engines (emphasis being on the strategies for suggesting alternative queries and the properties of these suggestions), we focused on no-answer queries (NAQs). We built machine-learning models to predict NAQs for mobile and meta-search scenarios, where such predictors may save various resources by preventing NAQs being submitted to the search engines. Our extensive experiments show that, although solving NAQs may be a difficult problem, their prediction is a relatively easy task.

In this paper, by characterizing NAQs, we pave the way to the development of more sophisticated techniques to solve them. When coupled with NAQ prediction, these techniques can be used without hurting the effectiveness and efficiency of other queries. In the future, we plan to investigate possible techniques for solving NAQs.

Acknowledgment

I. S. Altingovde is partially funded by Turkish Academy of Sciences (TUBA) Distinguished Young Scientist Award (GEBIP 2016).

References

Altingovde, I., Blanco, R., Cambazoglu, B.B., Ozcan, R., Sarigil, E., & Ulusoy, Ö. (2012). Characterizing web search queries that match very few or no results. In *Proceedings of 21st ACM CIKM* (pp. 2000–2004). New York: ACM.

Altingovde, I., Ozcan, R., & Ulusoy, Ö. (2011). Evolution of web search results within years. In *Proceedings of 34th International ACM SIGIR Conference* (pp. 1237–1238). New York: ACM.

Bailey, P., White, R.W., Liu, H., & Kumaran, G. (2010). Mining historic query trails to label long and rare search engine queries. ACM Transactions on the Web, 4, 15:1–15:27.

Balasubramanian, N., Kumaran, G., & Carvalho, V.R. (2010). Exploring reductions for long web queries. In *Proceedings of 33th International ACM SIGIR Conference* (pp. 571–578). New York: ACM.

Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D.A., & Frieder, O. (2004). Hourly analysis of a very large topically categorized web query log. In *Proceedings of 27th International ACM SIGIR Conference* (pp. 321–328). New York: ACM.

Bendersky, M., & Croft, W.B. (2009). Analysis of long queries in a large scale search log. In *Proceedings of 2009 Workshop on Web Search Click Data* (pp. 8–14). New York: ACM.

Broder, A.Z., Fontoura, M., Gabrilovich, E., Joshi, A., Josifovski, V., & Zhang, T. (2007). Robust classification of rare queries using web knowledge. In *Proceedings of 30th International ACM SIGIR Conference* (pp. 231–238). New York: ACM.

Cambazoglu, B.B., Altingovde, I.S., Ozcan, R., & Ulusoy, Ö. (2012). Cache-based query processing for search engines. ACM Transactions on the Web, 6, 14:1–14:24.

Carmel, D., Yom-Tov, E., Darlow, A., & Pelleg, D. (2006). What makes a query difficult? In *Proceedings of 29th International ACM SIGIR Conference* (pp. 390–397). New York: ACM.

Cucerzan, S., & Brill, E. (2004). Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics (ACL).

Downey, D., Dumais, S., & Horvitz, E. (2007). Heads and tails: Studies of web search with common and rare queries. In *Proceedings of 30th International ACM SIGIR Conference* (pp. 847–848). New York: ACM.

Friedman, J.H. (2000). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29, 1189–1232.

He, B., & Ounis, I. (2006). Query performance prediction. Information Systems, 31, 585–594.

Huston, S., & Croft, W.B. (2010). Evaluating verbose query processing techniques. In *Proceedings of 30th International ACM SIGIR Conference* (pp. 291–298). New York: ACM.

Jansen, B.J., Spink, A., & Koshman, S. (2007). Web searcher interaction with the dogpile.com metasearch engine. JASIST, 58, 744–755.

Jansen, B.J., Booth, D.L., & Spink, A. (2009). Patterns of query reformulation during Web searching. JASIST, 60, 1358–1371.

Jones, R., Rey, B., Madani, O., & Greiner, W. (2006). Generating query substitutions. In *Proceedings of 15th International Conference on WWW* (pp. 387–396). New York: ACM.

Kumaran, G., & Carvalho, V.R. (2009). Reducing long queries using query quality predictors. In *Proceedings of 32th International ACM SIGIR Conference* (pp. 564–571). New York: ACM.

Li, X., Schijvenaars, B.J.A., & de Rijke, M. (2017). Investigating queries and search failures in academic search. Information Processing and Management, 53, 666–683.

McCown, F., & Nelson, M.L. (2007). Search engines and their public interfaces: Which APIS are the most synchronized? In *Proceedings of 15th International Conference on WWW* (pp. 1197–1198). New York: ACM.

Pass, G., Chowdhury, A., & Torgeson, C. (2006). A picture of search. In *Proceedings of 1st International Conference on Scalable Information Systems* (p. 1). New York: ACM.

Pu, K.Q., & Yu, X. (2008). Keyword query cleaning. VLDB Endowment, 1, 909–920.

White, R.W., & Dumais, S.T. (2009). Characterizing and predicting search engine switching behavior. In *Proceedings of 18th ACM CIKM* (pp. 87–96). New York: ACM.

White, R.W., Richardson, M., Bilenko, M., & Heath, A.P. (2008). Enhancing web search by promoting multiple search engine use. In *Proceedings of 31st International ACM SIGIR Conference* (pp. 43–50). New York: ACM.

Zaragoza, H., Cambazoglu, B.B., & Baeza-Yates, R. (2010). Web search solved? All result rankings the same? In *Proceedings of 19th ACM CIKM* (pp. 529–538). New York: ACM.

Zheng, Z., Zha, H., Zhang, T., Chapelle, O., Chen, K., & Sun, G. (2007). A general boosting method and its application to learning ranking functions for web search. In *Proceedings of 21st Annual Conference on NIPS* (pp. 1697–1704). New York: Curran Associates, Inc.

DOI: 10.1002/asi